Trajectory Forecasting through Low-Rank Adaptation of Discrete Latent Codes

Riccardo Benaglia[1,2][0009−0002−8788−1156], Angelo Porrello[1][0000−0002−9022−8484], Pietro Buzzega[1][0000−0002−6516−6373], Simone Calderara[1][0000−0001−9056−1538], and Rita Cucchiara[1][0000−0002−2239−283X]

[1] AImageLab - University of Modena and Reggio Emilia, Modena, Italy
name.surname@unimore.it
[2] Ammagamma S.r.l.

Abstract. Trajectory forecasting is crucial for video surveillance analytics, as it enables the anticipation of future movements for a set of agents, *e.g.*, basketball players engaged in intricate interactions with long-term intentions. Deep generative models offer a natural learning approach for trajectory forecasting, yet they encounter difficulties in achieving an optimal balance between sampling fidelity and diversity. We address this challenge by leveraging Vector Quantized Variational Autoencoders (VQ-VAEs), which utilize a discrete latent space to tackle the issue of posterior collapse. Specifically, we introduce an instance-based codebook that allows tailored latent representations for each example. In a nutshell, the rows of the codebook are dynamically adjusted to reflect contextual information (*i.e.*, past motion patterns extracted from the observed trajectories). In this way, the discretization process gains flexibility, leading to improved reconstructions. Notably, instance-level dynamics are injected into the codebook through low-rank updates, which restrict the customization of the codebook to a lower dimension space. The resulting discrete space serves as the basis of the subsequent step, which regards the training of a diffusion-based predictive model. We show that such a two-fold framework, augmented with instance-level discretization, leads to accurate and diverse forecasts, yielding state-of-the-art performance on three established benchmarks.

Keywords: Trajectory forecasting · Vector Quantization.

1 Introduction

Trajectory forecasting finds applications in video surveillance [19], multi-object tracking [22, 6], behavioural analysis [29], and intrusion detection [34]. The goal is to predict the future paths of a set of agents from a few observations of their motion. The prediction can incorporate the interactions between pedestrians [15, 33, 24], or visual attributes of the environment they move within [5].

As multiple plausible paths can be forecast, trajectory prediction reveals an uncertain and multi-modal nature. To achieve this, recent data-driven approaches [23, 12, 16] lean toward a stochastic formulation that places a distribution over the future trajectory, rather than a single estimated path with 100%

certainty (*i.e.*, *deterministic* approaches [24]). In doing so, recent stochastic methods take advantage of the latest breakthroughs in deep generative modeling for image generation. For example, [12, 30] resorted to Generative Adversarial Networks, while [45, 16, 36] borrowed ideas from the class of variational methods.

One of the hindrances toward the application of variational approaches is the *posterior collapse* issue: *i.e.*, when the latent variables collapse to the prior becoming uninformative; as a consequence, the decoder learns to ignore them. This translates into a model with undermined generative capabilities, wherein its predictions are distributed on a single path (*e.g.*, , the most trivial one) with low uncertainty. A similar tendency (*mode collapse*) has been observed in adversarial networks, and has been addressed through burdensome learning objectives promoting variety [12, 30], or by devising multiple generator networks [5].

In the field of image generation, **Vector Quantized Variational Autoencoders [40]** (VQ-VAEs) have proven to mitigate posterior collapse. VQ-VAEs models avoid the hand-crafted Gaussian prior distribution; differently, they build upon a learnable categorical prior, thereby yielding a discrete latent space. The symbols of this space are the keys of a fixed-size dictionary (**codebook**), whose values are learnable latent codes. Thanks to the resulting increased flexibility, VQ-VAEs embody a promising paradigm for trajectory forecasting.

In this respect, our main contribution regards the content of the VQ-VAE codebook. In particular, while the original formulation devises a single codebook shared across all examples, we propose to dynamically adjust its values based on the *context* of each example, leading to an **instance-based** codebook. We refer as *context* to the set of historical information related to each agent, namely the past steps of its trajectory as well as its interactions with nearby agents. In this way, we aim to encourage even more flexibility during the discretization process, as distinct motion patterns can be discretized with varying granularity.

Moreover, we envision the customization of the codebook as an **adaptation** of the shared original VQ-VAE codebook. By doing so, our goal is to strike a balance between per-instance customization and the emergence of cross-instance concepts that are relevant across multiple examples. In practice, we draw inspiration from recent advances in Parameter Efficient Fine Tuning and represent the dynamic adjustments to the codebook as **low-rank updates** of its values (see Fig. 1). We show that such a modeling constraint improves the representation capabilities of the learned latent space, thereby encoding additional information and facilitating the reconstruction task. The traditional subsequent stage in VQ-VAEs involves fitting the distribution on the discrete latent codes. In this respect, we make use of a vector-quantized diffusion model [10] to learn the implicit prior, departing from existing approaches [40, 7] that rely on autoregressive priors, which are more susceptible to issues related to error accumulation.

The contributions are *i)* to the best of our knowledge, we are the first leveraging VQ-VAEs in a trajectory generation task; *ii)* we introduce a novel instance-based codebook based on low-rank modeling; *iii)* we achieve SOTA performance on three established benchmarks (Stanford Drone [28], NBA [20] and NFL [41]).

2 Related Work

The traditional approach to trajectory prediction considers solely the past movements of the agent [3]. However, its motion is likely to be influenced by the motions of other agents (*e.g.*, to avoid collisions or to perform coordinate actions). The first approaches took into account social behaviors through hand-crafted relations, energy-based features, or rule-based models [2, 26]. In recent years, the focus has shifted towards data-driven approaches [1, 12], leveraging deep models to extract social information [15, 33]. Others, instead, rely on the attention mechanism, which has proven highly effective at capturing interactions within tokenized data [24]. For example, [15] employs a graph-based attention mechanism to model human interactions, while [24] utilizes a social-temporal attention module to capture temporal relationships between consecutive time steps and interpersonal interactions occurring among agents.

Given the inherent uncertainty and multi-modal characteristics of future trajectories, recent approaches embrace a deep probabilistic framework to model their distribution. S-GAN [12] leverages a conditional Generative Adversarial Network (GAN) [8], while the authors of SoPhie [30] extend GANs to incorporates visual and social interaction components. Other works utilize conditional Variational Autoencoders (VAE) [17] for multimodal pedestrian trajectory prediction, including [45, 16, 36, 31, 46]. Trajectron++ [31] employs a VAE and represents agents' trajectories in a graph-structured recurrent neural network, while PECNet [23] integrates VAEs and goal conditioning. However, both GAN and VAE-based methods grapple with collapsing issues in trajectory generation, necessitating burdensome countermeasures [38]. Ultimately, the work by [11] pioneers the utilization of denoising diffusion models [13] within the trajectory prediction framework, marking a significant advancement in this domain.

Vector Quantization Models. Vector Quantized Variational Autoencoders [40] address posterior collapse by replacing the continuous latent space of VAEs with a discrete set of codewords. Starting from pioneering works, which showed the potential of these models in image generation [40, 27], recent studies focused on improving the two fundamental stages: the codebook learning and the discrete prior learning [18]. In this respect, SQ-VAE [35] replaces deterministic quantization with a pair of stochastic dequantization and quantization processes. To create a more comprehensive codebook, [7] supplements the original training losses of VQ-VAE with adversarial training. Additionally, [47] adopts a masking strategy during training and introduces prior distribution regularization to mitigate issues related to low-codebook utilization.

The advances regarding discrete prior learning involve architectural modifications [7] and a critical reevaluation of autoregression. [37] employs a discrete diffusion architecture to model code prediction, while MaskGIT [4] utilizes a bidirectional transformer decoder. This decoder generates all tokens of an image simultaneously and iteratively refines the image based on the preceding generation. In this paper, we condition the codebook on historical instance-level information while preserving the discrete nature of the latent space.

3 Preliminaries

We denote the future trajectory as $y \in \mathbb{R}^{T \times d}$, where T is the number of future time steps and d is the input channel dimension. When dealing with pedestrians, their trajectories are projected into the 2D bird's-eye view (so $d = 2$). The predicted trajectory $\hat{y}$ is generated by a learnable model, fed with a set of conditioning information: *i)* the observed trajectory $x \in \mathbb{R}^{T_p \times d}$ of the agent, *i.e.*, the coordinates observed at previous T_p steps, and *ii)* a set of neighboring trajectories denoted as $\mathcal{X} = \{x_1, x_2, \dots, x_N\}$. We define neighbors of an agent as all agents within the same scene, without imposing any distance threshold.

Vector Quantization. Standard VAEs [17] employ *i)* an **encoder** $\mathrm{E} \equiv \mathrm{E}(y|\theta_\mathrm{E})$ that, given input y, outputs a parametric posterior distribution $q(z|y)$ over latent variable z; *ii)* a **decoder** $\mathrm{G} \equiv \mathrm{G}(z|\theta_\mathrm{G})$ that provides the reconstruction of the input data as $p_{\theta_\mathrm{G}}(y|z)$. The posterior $q(z|y)$ is encouraged to conform to a standard Gaussian prior distribution $p(z)$, which could lead to over-regularized representations (posterior collapse). VQ-VAEs [40] extend VAEs by employing discrete latent variables and Vector Quantization (VQ) [9]. In particular, both posterior and prior distributions are categorical, and their samples provide indices for a learned **embedding table** $e \in \mathbb{R}^{C \times D}$, which consists of C static D-dimensional latent vectors. As outlined in the following paragraphs, the training of VQ-VAEs is divided into *learning the codebook* and *fitting the categorical prior*.

First Stage. Given the input $y \in \mathbb{R}^{T \times d}$, the encoder provides a continuous representation $z \in \mathbb{R}^{T \times D}$, where $z_t \in \mathbb{R}^D$ with $t \in \{1, 2, \dots, T\}$ and D indicates the dimension of the latent space. Then, the VQ-VAE characterizes the posterior as a joint distribution over T independent **categorical** variables $q(c_1, c_2, \dots, c_T|y)$ (one for each latent). Each marginal $q(c_t|y)$ is determined by matching each element of the encoding sequence z_t with the **nearest** vector in the codebook e:

$$q(c_t|y) = \underbrace{\mathcal{C}(p_1, p_2, \dots, p_C)}_{[0,\dots,0,1,0,\dots,0]} \text{ s.t. } p_c = \begin{cases} 1 & \text{if } c = \operatorname{argmin}_{c' \in \{1,2,\dots,C\}} \|z_t - e_{c'}\|_2^2 \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Notably, the posterior distribution is *deterministic* and not stochastic as for VAEs: hence, we can *draw* a sample $z^q \equiv z^q(y)$ from the posterior distribution by **selecting** the corresponding rows of the codebook, as follows:

$$\begin{gathered} z^q = [e_{c_1}, e_{c_2}, \dots, e_{c_T}] \\ c_t \sim q(c_t|y) \implies c_t = \operatorname{argmax} q(c_t|y). \end{gathered} \tag{2}$$

The subsequent step regards the decoder G, which reconstructs $\hat{y}$ from the sampled latent vector. During training, the first stage optimizes the following loss:

$$\mathcal{L}_\mathrm{FS} = \underbrace{\log p_{\theta_\mathrm{G}}(y|z^q)}_{\text{rec. error } e.g., \text{ MSE}} + \sum\nolimits_t \underbrace{\|\mathtt{sg}[z_t] - e_{c_t}\|^2}_{\text{embedding loss}} + \sum\nolimits_t \underbrace{\|z_t - \mathtt{sg}[e_{c_t}])\|^2}_{\text{commitment loss}}, \tag{3}$$

where `sg` is a shortcut for the `stopgradient` operator, which stops backpropagation from that computational node backward. The second term encourages the

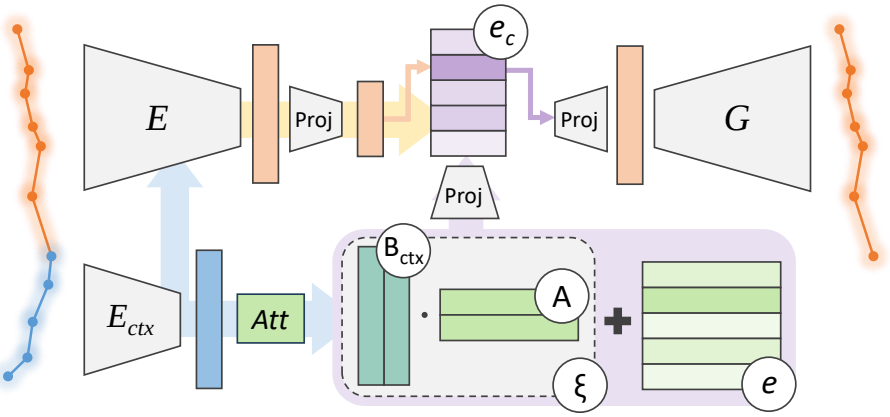

Fig. 1. Overview of our approach to trajectory prediction, based on Vector Quantization and Low-Rank adaptation of the codebook (highlighted in the purple box).

quantized latent vectors to be as close as possible to the nearest codeword, while the third one encourages the encoder to be *committed* to the chosen codeword.

Second Stage. The goal here is to learn a parametric model $p_{\theta_p}(c_1, c_2, \ldots, c_T)$ – termed *categorical prior* – which allows to draw new samples from the latent space. During this phase, the modules of the VQ-VAE are no longer subject to learning. Given the trained encoder, each training example y is embedded into a sequence of indices, built by relating each latent vector to the nearest row of the codebook (as in Eq. 2). On top of that, the generative model targets the generating process $p(c_1, c_2, \ldots, c_T)$ of the discrete latent codes, and optimizes the following Maximum Likelihood Estimation (MLE) training objective:

$$\mathcal{L}_{\text{SS}} = \mathbb{E}_{\substack{c_1,\ldots,c_T \\ c_t \sim q(c_t|y)}} [-\log p_{\theta_p}(c_1, c_2, \ldots, c_T)]. \tag{4}$$

4 Low-rank Adaptation for VQ-VAE

We herein present our approach to trajectory prediction, which we name LRVQ, depicted in Fig. 1. Briefly, we exploit VQ-VAEs to encode the future trajectory y of a given agent. On top of that, the following main novelties are introduced:

- We extend VQ-VAE to predict a trajectory coherent with the observed historical trend. To do so, we feed **additional contextual** information to the VQ-VAE, conditioning both the prior and the posterior distributions. The contextual information consists of the past observed trajectory x, and a summary of the interactions between the agent and its neighbours. The structure of the resulting quantization model is presented in Sec. 4.1.
- To encourage further **flexibility**, the codebook itself is conditioned on the additional contextual information (see Sec. 4.2). As discussed later, the context is introduced by devising a **low-rank** adjustment to the codebook.

– To avoid the error accumulation and the *unidirectional bias* problem, typical of auto-regressive methods [10], we make use of a **discrete diffusion model** for the generation of the sequence of indices (see Sec. 4.3). We also introduce a **new sampling technique**, based on the k-means clustering algorithm, to produce better and more consistent generations (see Sec. 4.4).

4.1 Trajectory Forecasting with VQ-VAEs

Formally, our VQ-VAE can be summarized as:

$$h_{\text{ctx}} = \text{E}_{\text{ctx}}([x, \mathcal{X}]) \qquad \text{(context encoding)} \tag{5a}$$

$$z^q = \text{E}(y, [\mathcal{Y}, h_{\text{ctx}}]) \qquad \text{(encoding)} \tag{5b}$$

$$\hat{y} = \text{G}(z^q, \mathcal{Z}^q), \qquad \text{(decoding)} \tag{5c}$$

where $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}^q$ represent respectively the past, the future, and the latent quantized representation of the nearby agents' trajectories (see Sec. 3). The modules $\text{E}_{\text{ctx}}(\cdot)$, $\text{E}(\cdot)$, $\text{G}(\cdot)$ are three neural networks, each of which exploits social-temporal transformer [24] to account for social-temporal relations.

In particular, a contextual encoder $\text{E}_{\text{ctx}}(\cdot)$ computes hidden features $h_{\text{ctx}} \in \mathbb{R}^{Tp \times D}$ that summarize both the past trend $x \in \mathbb{R}^{T_p \times 2}$ of the trajectory and spatial interactions (Eq. 5a). The function $\text{E}(\cdot)$ plays the role of the VQ-VAE encoder, transforming the future trajectory y into a discrete representation $z^q \in \mathbb{R}^{T \times D}$ (see Eq. 5b). To condition the model on historical information, the encoder is fed also with the hidden contextual information h_{ctx}; in detail, a tailored cross-attention layer is devised to mix future and past information. Finally, in step (5c) we achieve the estimated future trajectory $\hat{y} \in \mathbb{R}^{T \times 2}$ through the decoder $\text{G}(\cdot)$.

As well as traditional VQ-VAEs, we employ Mean Squared Error (MSE) as our reconstruction term between the ground truth and predicted trajectory.

4.2 Instance-based Codebook

The codebook plays a crucial role in VQ-VAEs and can cause instabilities during optimization. For instance, the uneven utilization of the vectors of the codebook is a factor that may lead to inefficiencies in representation learning. This imbalance often results in certain elements of the codebook being underutilized, while others never match with real-valued embeddings. To mitigate these issues, the authors of [44] resort to reducing the latent-space dimensionality, showing that it leads to a condensed but richer codebook. In practice, before quantization, each vector z is projected from $\mathbb{R}^D$ to a lower-dimension space $D_r \ll D$. In the following, we will refer to this strategy as **static codebook**, to distinguish it from our proposal that instead leverages dynamic cues.

Our idea is to modify the content of the codebook, such that it reflects the motion observed in the past trajectory. The intuition is that different motion styles (*e.g.*, straight *vs.* curvilinear) could prefer distinct latent codes and discretization strategies. On this basis, we exploit again the contextual features h_{ctx} to generate an **instance-based codebook** $\xi = f_\xi(\cdot, h_{\text{ctx}})$, computed through

a tailored learnable module f_ξ. The latter shares the same design of the above-described encoding networks and hence builds upon social-temporal transformers [24]. Afterwards, we combine static and instance-based codebooks by means of summation, thus obtaining a **conditioned** codebook e_c:

$$e_c = \texttt{l2\_norm}(e) + \lambda_\xi \texttt{l2\_norm}(\xi) \tag{6}$$

where `l2_norm` indicates the row-wise l2-normalization $v/\|v\|_2$ and λ_ξ is an hyper-parameter that weighs the sum. We leverage normalizing layers to ensure that the two components contribute almost equally to the final embedding table.

Moreover, the way we define the codebook draws inspiration from the successes of low-rank adaptation [14] for fine-tuning Large Language Models (LLMs). Namely, we opt for a *low-rank characterization* of f_ξ, which means that the instance-driven modifications to the static codebook lie on a lower-dimensional manifold of the parameter space. We hence define the instance-based codebook ξ as a matrix product of two low-rank matrices B_{ctx} and A, as follow:

$$\begin{aligned} B_{\text{ctx}} &= f_\xi(B, h_{\text{ctx}}) \quad \text{where } B, B_{\text{ctx}} \in \mathbb{R}^{D \times r} \\ \xi &= B_{\text{ctx}} A \quad \text{where } A \in \mathbb{R}^{r \times C}. \end{aligned} \tag{7}$$

Considering B as a set of learnable tokens, f_ξ adopts cross attention between the conditioning information h_{ctx} and B to create an instance-based B_{ctx}.

4.3 Diffusion-based Categorical Prior

As previously mentioned, the second main stage regards the training of the parametric categorical prior $p_{\theta_p}(c|x, \mathcal{X})$ (note that the p_{θ_p} is also conditioned on historical information), where $c = \{c_1, c_2, \ldots, c_T\}$. Notably, the learned prior serves to forecast the future trajectory y at inference time, when the posterior distribution of y is not available. Sec. 4.4 provides a detailed description of the sampling procedure, while the rest of this section describes the architectural and training aspects of the categorical prior.

We borrow the design of the categorical prior from the framework of Denoising Diffusion Probabilistic Models (DDPMs). In particular, we employ vector-quantized diffusion models [10], as they naturally handle discrete distributions. Notably, the application of DDPMs allows one to learn the categorical prior without the need for autoregressive modeling, as commonly employed in many existing approaches [39, 7]. In the context of trajectory prediction, we view the adoption of a non-autoregressive model as an additional strength. On the one hand, auto-regressive methods can leverage the inherent inductive bias of time-series data, where consecutive time steps relate to each other. However, this often results in error accumulation issues and in the so-called *unidirectional bias* [10], which blurs contextual information that flows in a direction not coherent with the chosen auto-regressive order. In the task under consideration, this means that auto-regressive approaches may struggle to leverage cues emerging in later moments of the trajectory, as *the goal* or the long-range intention of the agent. These crucial aspects of trajectory prediction [23] could be better addressed by

the approach proposed in this work, which is **order-free** and capable of capturing multiple plausible trends.

Formally, we define q^{diff} as the diffusion process that injects incremental noise to the token sequence c for Ψ diffusion steps. Instead, p_θ^{diff} is the denoising process that gradually reduces the noise of the noised sequence. The parameters θ of the denoising module are trained with the variational lower bound [32]:

$$\mathcal{L}_{\text{vlb}} = \mathcal{L}^0 + \mathcal{L}^1 + \cdots + \mathcal{L}^{\Psi-1} + \mathcal{L}^{\Psi}, \tag{8a}$$

$$\mathcal{L}^{\psi} = D_{KL}(q^{diff}(c^{\psi}|c^{\psi-1}) \parallel p_\theta^{diff}(c^{\psi}|c^{\psi+1}, \widehat{\mathcal{C}}^{\psi}, x, \mathcal{X})), \tag{8b}$$

$$\mathcal{L}^{c^0} = -\log p_\theta^{diff}(c^0|c^{\psi}, \widehat{\mathcal{C}}^{\psi}, x, \mathcal{X}), \tag{8c}$$

where we use x, $\mathcal{X}$ and $\widehat{\mathcal{C}}^{\psi}$ – the token sequence of neighboring agents at diffusion step ψ – as conditioning information during denoising. (8c) is an auxiliary objective encouraging the prediction of a noiseless token s_0. The loss function:

$$\mathcal{L} \leftarrow \begin{cases} \mathcal{L}^0, & \text{if } \psi = 1 \\ \mathcal{L}^{\psi-1} + \lambda \mathcal{L}^{c^0} & \text{otherwise.} \end{cases} \tag{9}$$

We refer to [10] for more exhaustive details on the diffusion steps and the prior.

Generation. At inference time, the past and social information is encoded using E_{ctx} and then passed to the diffusion process p_θ^{diff}. The latter, after Ψ denoising steps, provides a (denoised) sequence of T indices $\hat{c} \in \mathbb{R}^T$. These indices represent the encoding of the future unobserved trajectory; therefore, we used them to select the proper elements of the codebook e_c, thus allowing us to create a quantized sequence representation z^q. Then z^q undergoes decoding through the VQ-VAE decoder G, which finally yields the generation of trajectories $\hat{y}$.

4.4 Enforcing Effective Multi-modal Forecasting

The sampling approach described above represents the common way to draw new samples from the learned prior of a VQ-VAE. However, we build upon it to create a stronger and richer selection strategy that furthers the multi-modal capabilities of DDPMs. The standard evaluation process involves sampling K distinct trajectories from the model and assessing the top-performing one (as described in Sec. 5). Therefore, each methodology must find the right balance between accuracy in its prediction and potential for exploration. The proposed procedure goes in this direction: we generate numerous *raw* future paths, called *guesses*, and then condense them into the most representative ones. In formal terms, we sample N guesses and then perform the k-means clustering algorithm, with a number of clusters equal to $K < N$ (in our experiments, we set $N = 200$ and $K = 20$). We view the resulting *centroids* as the principal modes of the predictive distribution learned by the DDPM and thus use them for prediction in place of the original samples. This strategy guarantees a twofold advantage compared to naive prediction: firstly, out-of-distribution samples typically form independent clusters, thus enhancing exploration; secondly, the use of centroids reduces the quantization noise, as in-distribution samples are grouped into large clusters and averaged element-wise (see Fig. 2).

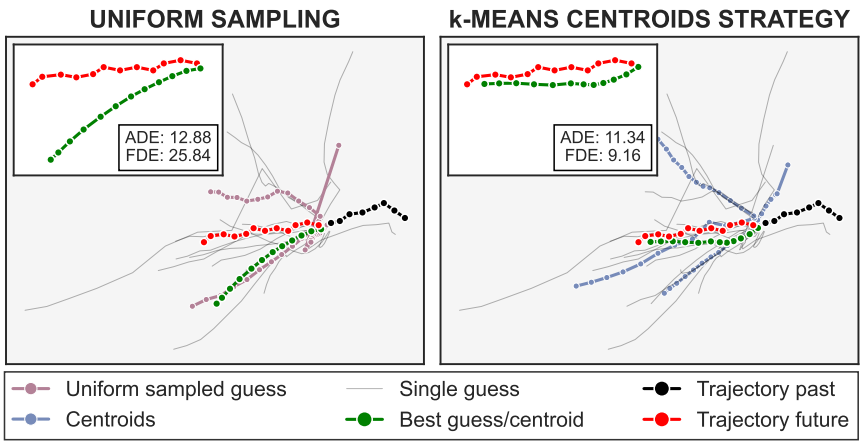

Fig. 2. Comparison between the $K = 5$ samples obtained from a uniform sampling strategy (on the left) and the ones given as output from the proposed k-means centroids sampling strategy (on the right), starting from the same $N = 20$ initial *guesses*.

5 Experiments

We assess our proposal on the following three trajectory prediction benchmarks.

Stanford Drone Dataset (SDD). The dataset [28] gathers trajectories of pedestrians within the Stanford University campus in a bird's eye view. Given 8 time steps (≈ 3.2 seconds), methods have to forecast the subsequent 12 frames (4.8 seconds). We employ the established train-test split [23].

NBA SportVU Dataset (NBA). Collected by the NBA's SportVU automatic tracking system, this dataset [20] provides the trajectories of 10 players and the ball in real basketball games. Given 10 previous time-steps (≈ 2.0 seconds), the models predict the subsequent 20 steps (4.0 seconds).

NFL Football Dataset (NFL). The NFL Football Dataset [41] records the movements of every player throughout each play of the 2017 season. The goal is to predict the trajectories of the 22 players (11 per team) and the ball for the ensuing 3.2 seconds (16 steps), given the preceding 1.6 seconds (8 steps).

Metrics. We use two established metrics [26, 1] *i.e.*, the Average/Final Displacement Errors (ADE/FDE). Given predicted and ground-truth trajectories, ADE computes the average error on all points, while the FDE restricts the error committed in the final step. Following other works dealing with stochastic models [16, 23], we adhere to the best-of-20 protocol [43, 42], selecting for evaluation the best trajectory from a pool of $K = 20$ generations. We denote the corresponding metrics as ADE_K and FDE_K; these are in meters for NBA and NFL, and in pixels for SDD. For sports datasets, we compute these metrics at different delta times to provide a more comprehensive assessment.

Table 1. Impact of distinct VQ-VAE codebooks on performance (ADE_{20}/FDE_{20}).

Dataset	Static	Full-Rank	Low-Rank
SDD	8.29/13.44	8.07/12.89	**7.86/12.68**
NBA	0.895/1.279	0.894/1.275	**0.893/1.267**
NFL	0.993/1.702	0.993/1.702	**0.982/1.679**

Implementation Details[3]**.** We set the number of codewords C to 16 for all datasets, while we take the best rank r for each dataset (*e.g.*, 8 for SDD and NBA, 4 for NFL). For the first stage, we use AdamW [21] as optimizer with $\text{lr} = 5 \times 10^{-4}$, $\beta_1 = 0.5$ and $\beta_2 = 0.9$. We train on SDD for 7000 epochs with batch size equal to 256. For NBA and NFL, we instead optimize for 700 epochs (the batch size equals 64). We use a cosine schedule for λ_ξ from an initial value of 0 to a final value of 1. In this way, we can introduce the instance-level codebook gradually during training.

For the second stage, we re-use the same optimizer/batch-size setup, while training for 3000 epochs for SDD, 1000 epochs for NBA, and 700 for NFL. As an augmentation technique, we rotate the trajectories by a random angle, ranging between 0 and θ_{max}. We set θ_{max} to 180° for the first stage, while we find it beneficial to adopt a lower value (5°) for the second stage.

5.1 On the Impact of the Instance-based Codebook

To assess the merits of our *low-rank* instance-based codebook, we herein empirically compare it with two alternative strategies. On the one hand, we devise a comparison with a *static* codebook (→ standard VQ-VAEs, lacking instance-level conditioning). Secondly, we contrast it with a *full-rank* codebook (which includes instance-level conditioning but lacks low-rank design constraints). To be more precise, the *full-rank* codebook is a baseline approach herein provided, which computes the values of the codebook through a learnable module fed with historical information as input. Unlike the proposed *low-rank* counterpart, the *full-rank* codebook does not adapt a shared static codebook but directly outputs its values. Through such a comparison, we can evaluate the efficacy of constraining the updates to the dictionary within a low-dimensional manifold.

Tab. 1 presents the related results: as can be observed, the *low-rank* model outperforms both the *static* and *full-rank* variants. In particular, the improvements are remarkable for SDD and NFL and more modest for NBA. Moreover, the presence of instance-level conditioning, common to *full-* and *low-* approaches, proves particularly beneficial for the SDD dataset, as demonstrated by the gap w.r.t. the static codebook (similar evidence emerges for the NBA dataset).

In the second place, we aim to investigate the impact of the rank r, which controls the dimension of the matrix B_{ctx} (*i.e.*, the degree of instance-level cues

[3] The code is available at https://github.com/aimagelab/LRVQ.

Table 2. Impact of varying the rank of B on the behavior of the model. Optimal performance (ADE_{20}) is achieved by identifying a sweet spot characterized by a low reconstruction error (ADE_{rec}) and a high accuracy in code prediction (Acc).

Dataset	Rank	ADE_{rec} ↓	Acc(%) ↑	ADE_{20} ↓
SDD	4	3.41	26.38	7.96
	16	2.97	22.20	8.06
NBA	4	0.207	15.92	0.898
	16	0.164	13.27	0.892
NFL	4	0.227	15.30	0.982
	16	0.177	11.95	0.996

introduced into the codebook). In particular, we want to measure how the rank r affects: *i)* the reconstruction capabilities of the VQ-VAE decoder (learned during the first stage); *ii)* the generative capabilities of the diffusion model (learned during the second stage). For point *i)*, we exploit the Average Displacement Error (ADE_{rec}) to assess the reconstruction performance. Instead, to characterize the generative capabilities, we resort to the mean accuracy achieved by the diffusion model in predicting codebook indexes, as well as the already mentioned ADE_{20}.

Tab. 2 presents the results for different ranks r. We observe that a higher reconstruction capability during the initial training stage is associated with increased difficulty in the diffusion task, resulting in lower accuracy. This indicates a correlation between the two phases: achieving optimal results in the first phase does not necessarily yield the best final generation metrics, as it complicates the joint task of trajectory generation (*i.e.*, sampling from the prior and reconstructing through the decoder). Tab. 2 demonstrates that the most favorable final metrics are achieved by striking a balance between low reconstruction error and good diffusion accuracy.

5.2 Comparison with SOTA Methods

In this section, we compare our model to the following existing approaches:

- Social-GAN [12] relies on a Conditioned GAN, with a module to handle social interactions between agents.
- Trajectron++ [31] exploits VAEs and graph-structured recurrent networks.
- PECNet [23] augments a VAE with *goal-oriented* reasoning.
- LB-EBM [25] targets the prediction of long-range trajectories through a belief vector, which encapsulates the energy distribution in the environment.
- GroupNet [42] is a multiscale hypergraph network that captures both pair- and group-wise interactions at different scales.
- Memo-Net [43] mimics retrospective memory in neuropsychology and predicts intentions by retrieving similar instances from a memory bank.
- MID [11] leverages a diffusion model to progressively reduce indeterminacy within potential future paths.

Table 3. SDD results (ADE_{20}/FDE_{20}). * represents the reproduced results from open source. Best results in **bold**, second-best underlined.

Time	S-GAN	Trajectron++	PECNet	MemoNet	GroupNet	MID*	**LRVQ**
4.8s	27.23/41.44	19.30/32.70	9.96/15.88	8.56/**12.66**	9.31/16.11	9.73/15.32	**7.86**/12.68

Table 4. NBA results (ADE_{20}/FDE_{20}). Best results in **bold**, second-best underlined.

Time	S-GAN	PECNet	Trajectron++	MemoNet	GroupNet	MID	**LRVQ**
1.0s	0.41/0.62	0.40/0.71	0.30/0.38	0.38/0.56	0.26/0.34	0.28/0.37	**0.19**/**0.29**
2.0s	0.81/1.32	0.83/1.61	0.59/0.82	0.71/1.14	0.49/0.70	0.51/0.72	**0.41**/**0.63**
3.0s	1.19/1.94	1.27/2.44	0.85/1.24	1.00/1.57	0.73/1.02	0.71/0.98	**0.64**/**0.96**
4.0s	1.59/2.41	1.69/2.95	1.15/1.57	1.25/1.47	0.96/1.30	0.96/**1.27**	**0.89**/**1.27**

Table 5. NFL results (ADE_{20}/FDE_{20}). Best results in **bold**, second-best underlined.

Time	S-GAN	PECNet	Trajectron++	LB-EBM	GroupNet	MID	**LRVQ**
1.0s	0.37/0.68	0.52/0.97	0.41/0.65	0.75/1.05	0.32/0.57	0.30/0.58	**0.23**/**0.35**
2.0s	0.83/1.53	1.19/2.47	0.93/1.65	1.26/2.28	0.73/1.39	0.71/1.31	**0.53**/**0.92**
3.2s	1.44/2.51	1.99/3.84	1.54/2.58	1.90/3.25	1.21/2.15	1.14/1.92	**0.98**/**1.68**

We report the comparison in Tab. 3, Tab. 4, and Tab. 5. To sum up, our LRVQ demonstrates superior performance across all the considered benchmarks.

On the SDD dataset (Tab. 3), we attain superior ADE results, matching closely MemoNet in FDE. While PECNet and GroupNet, among C-VAE methods, demonstrate noteworthy performance compared to the older S-GAN and Trajectron++, they struggle in FDE, especially when compared to MemoNet. This could be ascribed to the effective sampling strategy of MemoNet, which integrates a tailored clustering phase to generate multiple overall intentions.

Additionally, our approach showcases robust performance across all examined partial timestamps for both the NBA (Tab. 4) and NFL datasets (Tab. 5). The two most competing methods are GroupNet – based on the C-VAE framework – and more importantly MID, which akin to our approach utilizes a diffusion process. However, we highlight an important distinction with MID, which we consider as a motivation for our improvements: while MID adopts diffusion modeling directly in output space, we instead apply it to the discrete variables extracted by the VQ-VAE encoder. We believe that our latent-based formulation further promotes the emergence of multi-modal generative capabilities.

5.3 Qualitative Results

Figure 3 provides a qualitative comparison on 20 generations (with sub-sampling) produced by a VQ-VAE trained with a *static* codebook, a *dynamic* codebook, and the *low-rank* conditioned codebook (see Sec. 5.1). Each row illustrates a different scene from the SDD dataset, showcasing different agent behaviors: in

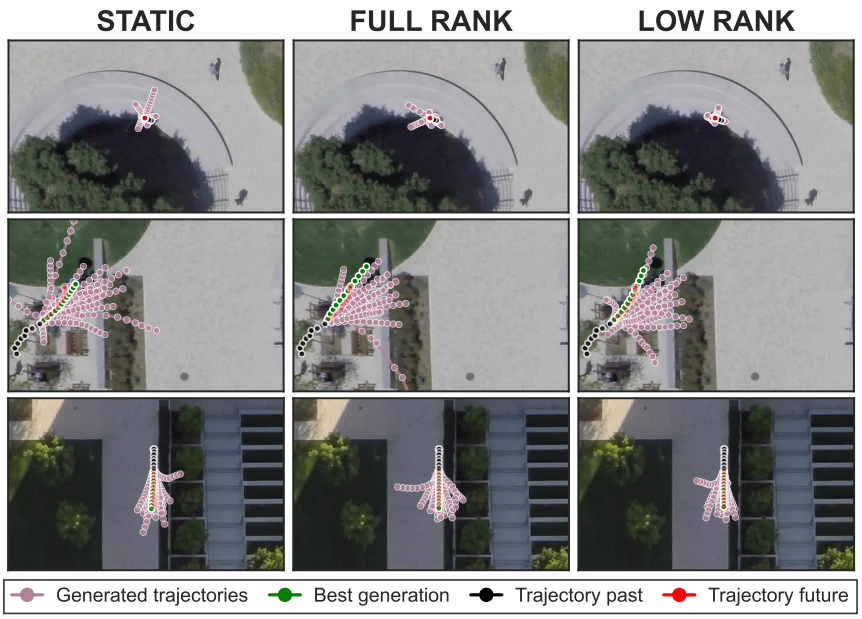

Fig. 3. Qualitative comparison for three SDD scenes (one for each row of the figure) between the trajectories obtained from a VQ-VAE with a static codebook, a full rank codebook the proposed *low-rank* codebook (from left to right).

the first one, the agent remains stationary, while in the others, it either turns left or proceeds straight ahead. Compared to the other two methods, low-rank conditioning appears to be more accurate, particularly in complex scenarios where the agent stays still or changes its direction of movement.

6 Discussion and Conclusions

Limitations. The complexity of our model is linked to two factors:

- Two-step training procedure: although VQ-VAE offers benefits such as a learned prior, the training must be divided into two distinct stages, which increases the total time required to train the model.
- Inference time: the inference procedure described in Sec. 4.4 takes longer as the number N of starting guesses increases. To obtain a trade-off between the accuracy of the ensemble of K final generations and the computational time, the parameter N has to be carefully adjusted.

Conclusion. We propose a stochastic approach for trajectory prediction. It builds upon Vector Quantization to yield a predictive distribution that preserves both sampling fidelity and diversity. Our main contribution lies in a dynamic, instance-related codebook encompassing past trajectory information. Notably,

contextual information is incorporated into the codebook through a low-rank update. We conduct several empirical studies to validate our approach, demonstrating its superior generative capabilities compared to both standard VQ-VAEs and existing methods. This leads to state-of-the-art results on three established benchmarks.

Acknowledgement

The research was supported by the Italian Ministry for University and Research through the PNRR project ECOSISTER ECS 00000033 CUP E93C22001100001 and by the EU Horizon project "ELIAS - European Lighthouse of AI for Sustainability" (No. 101120237).

References

1. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social lstm: Human trajectory prediction in crowded spaces. In: CVPR (2016)
2. Antonini, G., Bierlaire, M., Weber, M.: Discrete choice models of pedestrian walking behavior. Transp. Res. B Methodol **40** (2006)
3. Becker, S., Hug, R., Hubner, W., Arens, M.: Red: A simple but effective baseline predictor for the trajnet benchmark. In: ECCVW (2018)
4. Chang, H., Zhang, H., Jiang, L., Liu, C., Freeman, W.T.: Maskgit: Masked generative image transformer. In: CVPR (2022)
5. Dendorfer, P., Elflein, S., Leal-Taixé, L.: Mg-gan: A multi-generator model preventing out-of-distribution samples in pedestrian trajectory prediction. In: ICCV (2021)
6. Dendorfer, P., Yugay, V., Osep, A., Leal-Taixé, L.: Quo vadis: Is trajectory forecasting the key towards long-term multi-object tracking? Advances in Neural Information Processing Systems **35**, 15657–15671 (2022)
7. Esser, P., Rombach, R., Ommer, B.: Taming transformers for high-resolution image synthesis. In: CVPR (2021)
8. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. NeurIPS (2014)
9. Gray, R.M., Neuhoff, D.L.: Quantization. IEEE Trans. Inf. Theory **44** (1998)
10. Gu, S., Chen, D., Bao, J., Wen, F., Zhang, B., Chen, D., Yuan, L., Guo, B.: Vector quantized diffusion model for text-to-image synthesis. In: CVPR (2022)
11. Gu, T., Chen, G., Li, J., Lin, C., Rao, Y., Zhou, J., Lu, J.: Stochastic trajectory prediction via motion indeterminacy diffusion. In: CVPR (2022)
12. Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., Alahi, A.: Social gan: Socially acceptable trajectories with generative adversarial networks. In: CVPR (2018)
13. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. NeurIPS (2020)
14. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models. ICLR (2021)
15. Huang, Y., Bi, H., Li, Z., Mao, T., Wang, Z.: Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In: ICCV (2019)
16. Ivanovic, B., Pavone, M.: The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In: ICCV (2019)

17. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. ICLR (2014)
18. Kolesnikov, A., Susano Pinto, A., Beyer, L., Zhai, X., Harmsen, J., Houlsby, N.: Uvim: A unified modeling approach for vision with learned guiding codes. NeurIPS (2022)
19. Li, Y., Liang, R., Wei, W., Wang, W., Zhou, J., Li, X.: Temporal pyramid network with spatial-temporal attention for pedestrian trajectory prediction. IEEE TNSE (2021)
20. linouk23: Nba player movements. https://github.com/linouk23/NBA-Player-Movements, accessed: 2016
21. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. ICLR (2019)
22. Mancusi, G., Panariello, A., Porrello, A., Fabbri, M., Calderara, S., Cucchiara, R.: Trackflow: Multi-object tracking with normalizing flows. In: ICCV (2023)
23. Mangalam, K., Girase, H., Agarwal, S., Lee, K.H., Adeli, E., Malik, J., Gaidon, A.: It is not the journey but the destination: Endpoint conditioned trajectory prediction. In: ECCV (2020)
24. Monti, A., Porrello, A., Calderara, S., Coscia, P., Ballan, L., Cucchiara, R.: How many observations are enough? knowledge distillation for trajectory forecasting. In: CVPR (2022)
25. Pang, B., Zhao, T., Xie, X., Wu, Y.N.: Trajectory prediction with latent belief energy-based model. In: CVPR (2021)
26. Pellegrini, S., Ess, A., Schindler, K., Van Gool, L.: You'll never walk alone: Modeling social behavior for multi-target tracking. In: ICCV (2009)
27. Razavi, A., Van den Oord, A., Vinyals, O.: Generating diverse high-fidelity images with vq-vae-2. NeurIPS (2019)
28. Robicquet, A., Sadeghian, A., Alahi, A., Savarese, S.: Learning social etiquette: Human trajectory understanding in crowded scenes. In: ECCV (2016)
29. Rudenko, A., Palmieri, L., Herman, M., Kitani, K.M., Gavrila, D.M., Arras, K.O.: Human motion trajectory prediction: A survey. IJRR **39** (2020)
30. Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Rezatofighi, H., Savarese, S.: Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In: CVPR (2019)
31. Salzmann, T., Ivanovic, B., Chakravarty, P., Pavone, M.: Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In: ECCV (2020)
32. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: ICML (2015)
33. Sun, C., Karlsson, P., Wu, J., Tenenbaum, J.B., Murphy, K.: Stochastic prediction of multi-agent interactions from partial observations. ICLR (2019)
34. Sun, J., Chen, J., Chen, T., Fan, J., He, S.: Pidnet: An efficient network for dynamic pedestrian intrusion detection. In: ACM Multimedia (2020)
35. Takida, Y., Shibuya, T., Liao, W., Lai, C.H., Ohmura, J., Uesaka, T., Murata, N., Takahashi, S., Kumakura, T., Mitsufuji, Y.: Sq-vae: Variational bayes on discrete representation with self-annealed stochastic quantization. ICML (2022)
36. Tang, C., Salakhutdinov, R.R.: Multiple futures prediction. NeurIPS (2019)
37. Tang, Z., Gu, S., Bao, J., Chen, D., Wen, F.: Improved vector quantized diffusion models. arXiv preprint (2022)
38. Thiede, L.A., Brahma, P.P.: Analyzing the variety loss in the context of probabilistic trajectory prediction. In: ICCV (2019)
39. Van Den Oord, A., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks. In: ICML (2016)

40. Van Den Oord, A., Vinyals, O., et al.: Neural discrete representation learning. NeurIPS **30** (2017)
41. a vhadgar: Big data bowl. https://github.com/a-vhadgar/Big-Data-Bowl, accessed: 2017
42. Xu, C., Li, M., Ni, Z., Zhang, Y., Chen, S.: Groupnet: Multiscale hypergraph neural networks for trajectory prediction with relational reasoning. In: CVPR (2022)
43. Xu, C., Mao, W., Zhang, W., Chen, S.: Remember intentions: Retrospective-memory-based trajectory prediction. In: CVPR (2022)
44. Yu, J., Li, X., Koh, J.Y., Zhang, H., Pang, R., Qin, J., Ku, A., Xu, Y., Baldridge, J., Wu, Y.: Vector-quantized image modeling with improved vqgan. ICLR (2022)
45. Yuan, Y., Kitani, K.: Diverse trajectory forecasting with determinantal point processes. ICLR (2020)
46. Yuan, Y., Weng, X., Ou, Y., Kitani, K.M.: Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In: ICCV (2021)
47. Zhang, J., Zhan, F., Theobalt, C., Lu, S.: Regularized vector quantization for tokenized image synthesis. In: CVPR (2023)